

# Ensemble Learning

---

Nipun Batra and teaching staff

IIT Gandhinagar

July 30, 2025

Based on [Ensemble methods in ML by Dietterich](#)

Three reasons why ensembles make sense:

Based on [Ensemble methods in ML by Dietterich](#)

Three reasons why ensembles make sense:

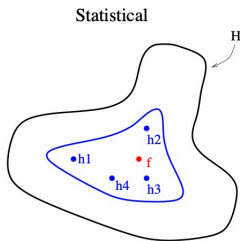
1) Statistical: Sometimes if **data is limited, many competing hypotheses can be learned** all giving the same accuracy on training data.

## Based on Ensemble methods in ML by Dietterich

Three reasons why ensembles make sense:

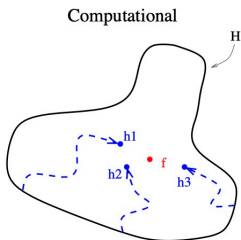
1) Statistical: Sometimes if **data is limited, many competing hypotheses can be learned** all giving the same accuracy on training data.

E.g., we can learn many decision trees for the same data giving the same accuracy.



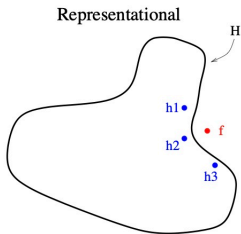
2) Computational: Even if data is sufficient, some **classifiers/regressors can get stuck in local optima or apply greedy strategies**. Computationally learning the "best" hypothesis can be non-trivial.

2) Computational: Even if data is sufficient, some **classifiers/regressors can get stuck in local optima or apply greedy strategies**. Computationally learning the "best" hypothesis can be non-trivial. E.g., decision trees employ greedy criteria



3) Representational: Some **classifiers/regressors cannot learn the true form or representation.**

3) Representational: Some **classifiers/regressors cannot learn the true form or representation**.  
E.g., decision trees can only learn axis-parallel splits.





1) A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse.

1) A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse.

2) An accurate classifier:

1) A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse.

2) An accurate classifier: is one that has an error rate of better than random guessing on new  $x$  values.

- 1) A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse.
- 2) An accurate classifier: is one that has an error rate of better than random guessing on new  $x$  values.
- 3) Two classifiers are diverse:

- 1) A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse.
- 2) An accurate classifier: is one that has an error rate of better than random guessing on new  $x$  values.
- 3) Two classifiers are diverse: if they make different errors on new data points

If the three classifiers are identical, i.e. not diverse, then when  $h_1(x)$  is wrong  $h_2(x)$  and  $h_3(x)$  will also be wrong.

If the three classifiers are identical, i.e. not diverse, then when  $h_1(x)$  is wrong  $h_2(x)$  and  $h_3(x)$  will also be wrong. However, if the errors made by the classifiers are uncorrelated, then when  $h_1(x)$  is wrong,  $h_2(x)$  and  $h_3(x)$  may be correct, so that a majority vote will correctly classify.

Error Probability of each model =  $\varepsilon = 0.3$

$$\begin{aligned} Pr(\text{ensemble being wrong}) &= \\ {}^3C_2(\varepsilon^2)(1 - \varepsilon)^{3-2} &+ {}^3C_3(\varepsilon^3)(1 - \varepsilon)^{3-3} \\ &= 0.19 \leq 0.3 \end{aligned}$$



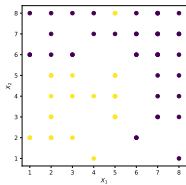




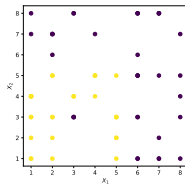




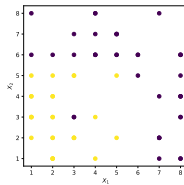
### Round - 1



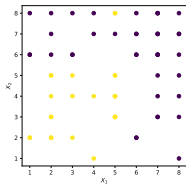
### Round - 2



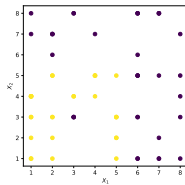
### Round - 3



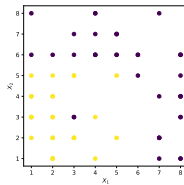
# Round - 1



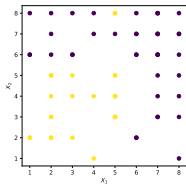
# Round - 2



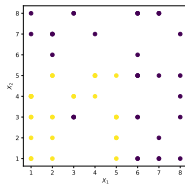
# Round - 3



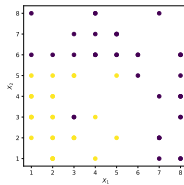
# Round - 1



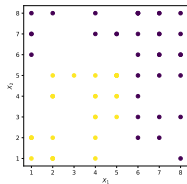
# Round - 2



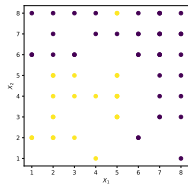
# Round - 3



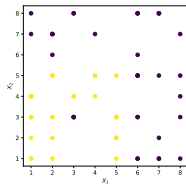
# Round - 4



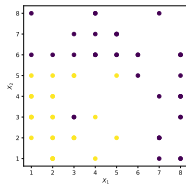
## Round - 1



## Round - 2

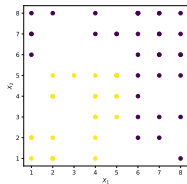


## Round - 3

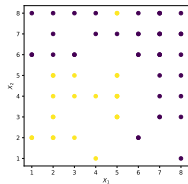




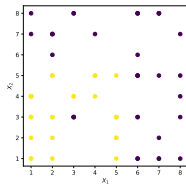
# Round - 4



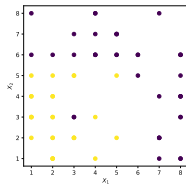
## Round - 1



## Round - 2



## Round - 3



Number of Clusters ( $X_1$ )	Number of Points
1	1
1	2
1	3
1	4
1	5
1	6
1	7
1	8
2	1
2	2
2	3
2	4
2	5
2	6
2	7
2	8
3	1
3	2
3	3
3	4
3	5
3	6
3	7
3	8
4	1
4	2
4	3
4	4
4	5
4	6
4	7
4	8
5	1
5	2
5	3
5	4
5	5
5	6
5	7
5	8
6	1
6	2
6	3
6	4
6	5
6	6
6	7
6	8
7	1
7	2
7	3
7	4
7	5
7	6
7	7
7	8
8	1
8	2
8	3
8	4
8	5
8	6
8	7
8	8

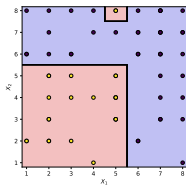






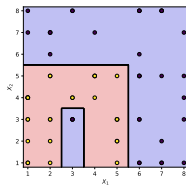


Round - 1



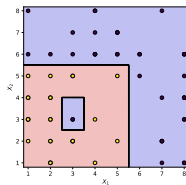
Tree Depth = 4

Round - 2



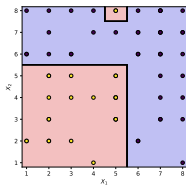
Tree Depth = 5

Round - 3



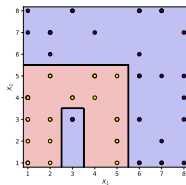
Tree Depth = 5

Round - 1



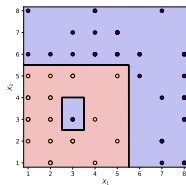
Tree Depth = 4

Round - 2



Tree Depth = 5

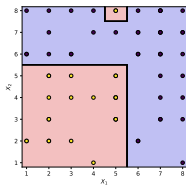
Round - 3



Tree Depth = 5

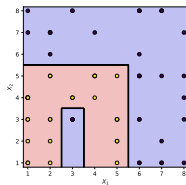


Round - 1



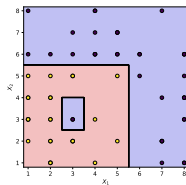
Tree Depth = 4

Round - 2



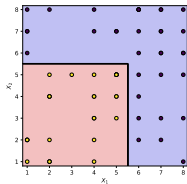
Tree Depth = 5

Round - 3

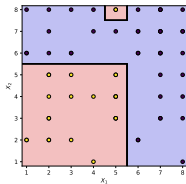


Tree Depth = 5

Round - 4

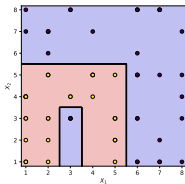


Tree Depth = 2  
Round - 1



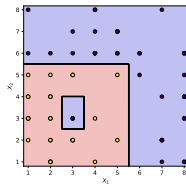
Tree Depth = 4

Round - 2



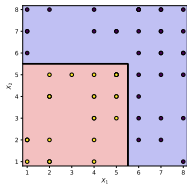
Tree Depth = 5

Round - 3



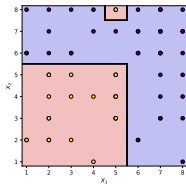
Tree Depth = 5

Round - 4



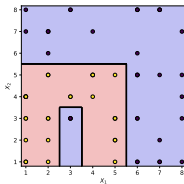
Tree Depth = 2

Round - 1



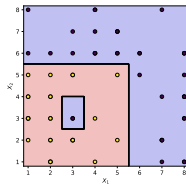
Tree Depth = 4

Round - 2



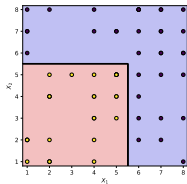
Tree Depth = 5

Round - 3

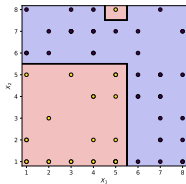


Tree Depth = 5

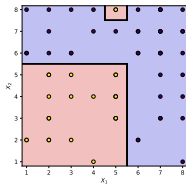
Round - 4



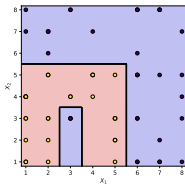
Round - 5



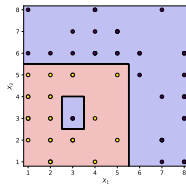
Tree Depth = 2  
Round - 1



Round - 2



Tree Depth = 4  
Round - 3



Tree Depth = 4

Tree Depth = 5

Tree Depth = 5

All learners are incrementally built.

All learners are incrementally built.

All learners are incrementally built.

Incremental building: Incrementally try to classify "harder" samples correctly.

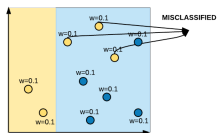




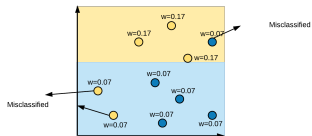




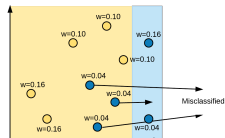




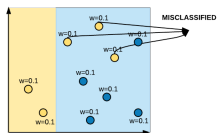
$$\alpha_1 = 0.42$$



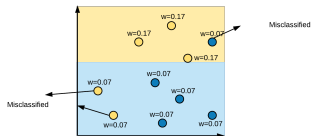
$$\alpha_2 = 0.66$$



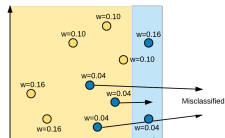
$$\alpha_3 = 0.99$$



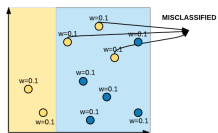
$$\alpha_1 = 0.42$$



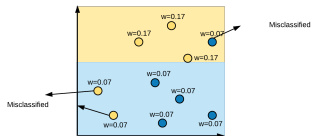
$$\alpha_2 = 0.66$$



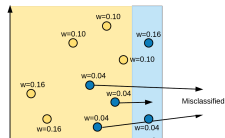
$$\alpha_3 = 0.99$$



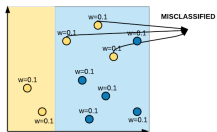
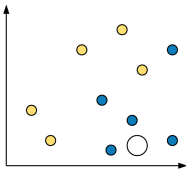
$$\alpha_1 = 0.42$$



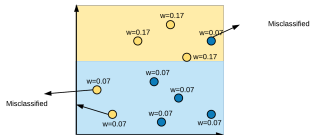
$$\alpha_2 = 0.66$$



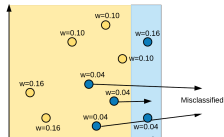
$$\alpha_3 = 0.99$$



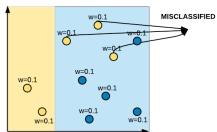
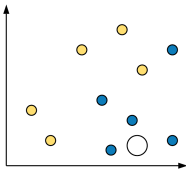
$$\alpha_1 = 0.42$$



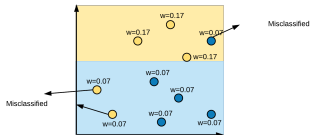
$$\alpha_2 = 0.66$$



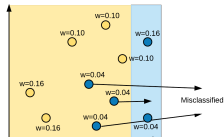
$$\alpha_3 = 0.99$$



$$\alpha_1 = 0.42$$

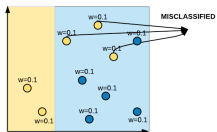
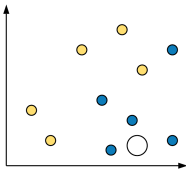


$$\alpha_2 = 0.66$$

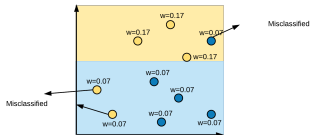


$$\alpha_3 = 0.99$$

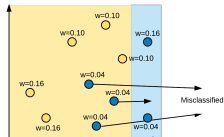




$$\alpha_1 = 0.42$$

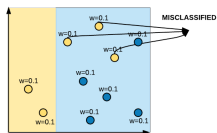
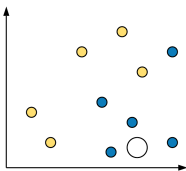


$$\alpha_2 = 0.66$$

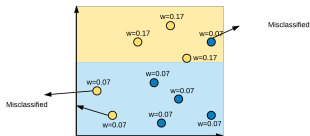


$$\alpha_3 = 0.99$$

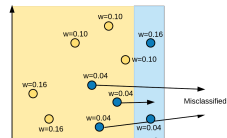
Let us say, yellow class is +1  
and blue class is -1



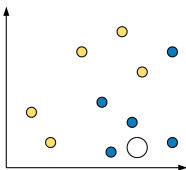
$$\alpha_1 = 0.42$$



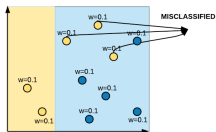
$$\alpha_2 = 0.66$$



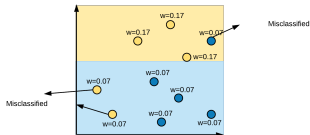
$$\alpha_3 = 0.99$$



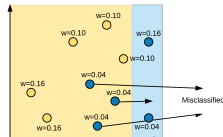
Let us say, yellow class is +1  
and blue class is -1  
Prediction =  $\text{SIGN}(0.42 \cdot -1 + 0.66 \cdot -1 + 0.99 \cdot +1)$  =  
Negative = blue



$$\alpha_1 = 0.42$$



$$\alpha_2 = 0.66$$

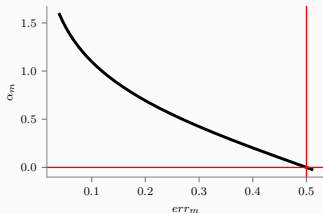


$$\alpha_3 = 0.99$$

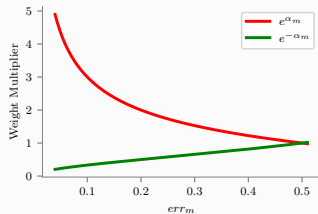




## Notebook: [boosting-explanation.html](#)



## Notebook: [boosting-explanation.html](#)



# ADABOOST for regression

## From Paper: Improving Regressors using Boosting Techniques

Our problem will be that the modeling error is also nonzero because we have to determine the model in the presence of noise. Since we don't know the probability distributions, we approximate the expectation of the ME and PE using the sample ME (if the truth is known) and sample PE and then average over multiple experiments.

In the following discussion, we detail both bagging and boosting. We then discuss how to build trees which are the basic building blocks of our regression machines and use these ensembles on some standard test functions.

### 2. BAGGING

The following is a paraphrase of Breiman (1996b) with some difference in notation. Suppose we pick with replacement  $N_1$  examples from the training set of size  $N_1$  and call the  $k$ 'th set of observations  $O_k$ . Based on these observations, we form a predictor  $y^{(p)}(\mathbf{x}, O_k)$ . Because we are sampling with replacement, we may have multiple observations or no observations of a particular training example. Sampling with replacement is sometimes termed bootstrap sampling [Efron and Tibshirani (1993)] and therefore this method is called bootstrap aggregating or bagging for short. The ensemble predictor is formed from the approximation to the expectation over all the observation sets, i.e.  $E_O[y^{(p)}(\mathbf{x}, O)]$  by using the average of the outputs of all the predictors. Breiman discusses which algorithms are good candidates for predictors and concludes that the best predictors are unstable, i.e., a small change in the training set  $O_k$  causes a large change in the predictor  $y^{(p)}(\mathbf{x}, O_k)$ . Good candidates are regression trees and neural nets.

### 3. BOOSTING

In bagging, each training example is equally likely to be picked. In boosting, the probability of a particular example being in the training set of a particular machine depends on the performance of the prior machines on that example. The following is a modification of *Adaboost.R* [Freund and Schapire (1996a)].

Initially, to each training pattern we assign a weight  $w_i = 1 \quad i = 1, \dots, N_1$

Repeat the following while the average loss  $\bar{L}$  defined

set. Each machine makes a hypothesis:  $h_i: \mathbf{x} \rightarrow y$

3. Pass every member of the training set through this machine to obtain a prediction  $y_i^{(p)}(\mathbf{x}_i) \quad i = 1, \dots, N_1$ .

4. Calculate a loss for each training sample  $L_i = L \left[ \left| y_i^{(p)}(\mathbf{x}_i) - y_i \right| \right]$ . The loss  $L$  may be of any functional form as long as  $L \in [0, 1]$ . If we let

$$D = \sup \left| y_i^{(p)}(\mathbf{x}_i) - y_i \right| \quad i = 1, \dots, N_1$$

then we have three candidate loss functions:

$$L_i = \frac{\left| y_i^{(p)}(\mathbf{x}_i) - y_i \right|}{D} \quad (\text{linear})$$

$$L_i = \frac{\left| y_i^{(p)}(\mathbf{x}_i) - y_i \right|^2}{D^2} \quad (\text{square law})$$

$$L_i = 1 - \exp \left[ \frac{- \left| y_i^{(p)}(\mathbf{x}_i) - y_i \right|}{D} \right] \quad (\text{exponential})$$

5. Calculate an average loss:  $\bar{L} = \sum_{i=1}^{N_1} L_i p_i$

6. Form  $\beta = \frac{\bar{L}}{1 - \bar{L}}$ .  $\beta$  is a measure of confidence in the predictor. Low  $\beta$  means high confidence in the prediction.

7. Update the weights:  $w_i \rightarrow w_i \beta^{**} [1 - L_i]$ , where  $**$  indicates exponentiation. The smaller the loss, the more the weight is reduced making the probability smaller that this pattern will be picked as a member of the training set for the next machine in the ensemble.

8. For a particular input  $\mathbf{x}_i$ , each of the  $T$  machines makes a prediction  $h_t, t = 1, \dots, T$ . Obtain the cumulative prediction  $h_T$  using the  $T$  predictors:

# Random Forest

- Random Forest is an ensemble of decision trees.



# Random Forest

- Random Forest is an ensemble of decision trees.
- We have two types of bagging: bootstrap (on data) and random subspace (of features).

# Random Forest

- Random Forest is an ensemble of decision trees.
- We have two types of bagging: bootstrap (on data) and random subspace (of features).

# Random Forest

- Random Forest is an ensemble of decision trees.
- We have two types of bagging: bootstrap (on data) and random subspace (of features).
- As features are randomly selected, we learn decorrelated trees and helps in reducing variance.

# Random Forest

There are 3 parameters while training a random forest  
number of trees, number of features ( $m$ ), maximum  
depth.

## Training Algorithm

- For  $i^{th}$  tree ( $i \in \{1 \dots N\}$ ), select  $n$  samples from total  $N$  samples with replacement.

# Random Forest

There are 3 parameters while training a random forest  
number of trees, number of features ( $m$ ), maximum  
depth.

## Training Algorithm

- For  $i^{th}$  tree ( $i \in \{1 \dots N\}$ ), select  $n$  samples from total  $N$  samples with replacement.

# Random Forest

There are 3 parameters while training a random forest  
number of trees, number of features ( $m$ ), maximum  
depth.

## Training Algorithm

- For  $i^{th}$  tree ( $i \in \{1 \dots N\}$ ), select  $n$  samples from total  $N$  samples with replacement.
- Learn Decision Tree on selected samples for  $i^{th}$  round.

## Learning Decision Tree (for RF)

# Random Forest

There are 3 parameters while training a random forest  
number of trees, number of features ( $m$ ), maximum  
depth.

## Training Algorithm

- For  $i^{th}$  tree ( $i \in \{1 \dots N\}$ ), select  $n$  samples from total  $N$  samples with replacement.
- Learn Decision Tree on selected samples for  $i^{th}$  round.

## Learning Decision Tree (for RF)

- For each split, select  $m$  features from total available  $M$  features and train a decision tree on selected features

# Dataset

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

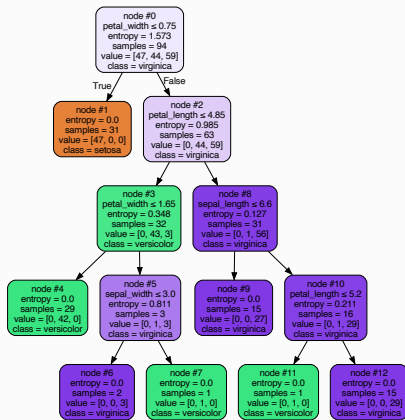
150 rows × 5 columns



for *depth* in  $[1, \dots, \textit{maximum depth}]$

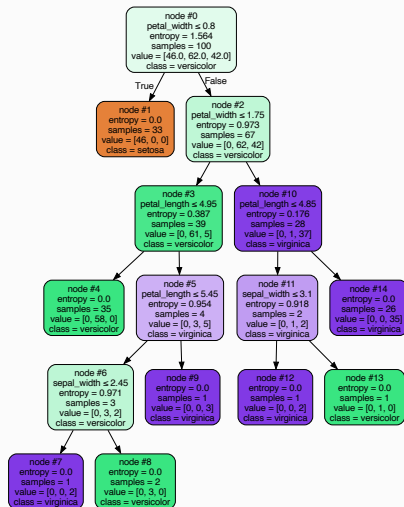
# Decision Tree # 0

**Notebook:** ensemble-feature-importance.html



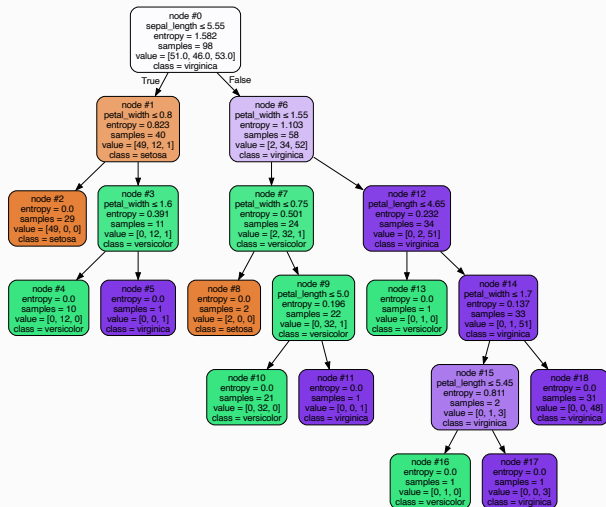
# Decision Tree # 1

## Notebook: ensemble-feature-importance.html



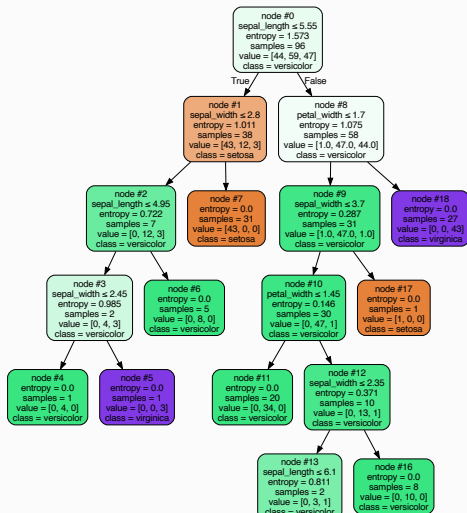
# Decision Tree # 2

## Notebook: ensemble-feature-importance.html



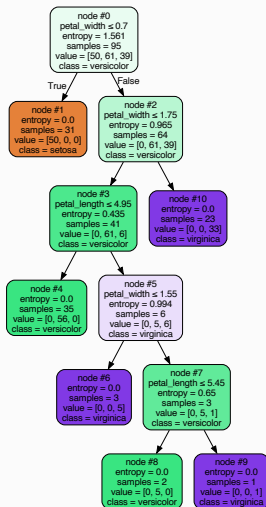
# Decision Tree # 3

## Notebook: ensemble-feature-importance.html



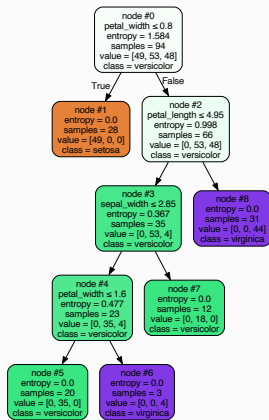
# Decision Tree # 4

## Notebook: ensemble-feature-importance.html



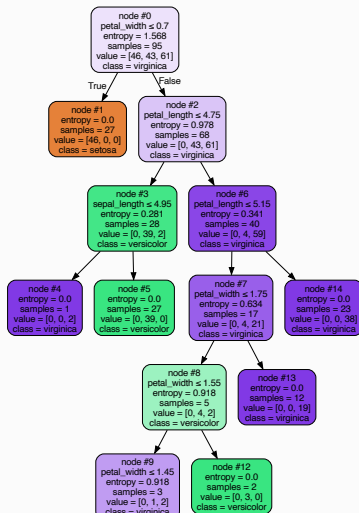
# Decision Tree # 5

**Notebook:** ensemble-feature-importance.html



# Decision Tree # 6

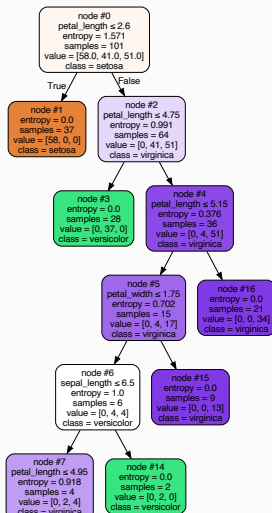
**Notebook:** ensemble-feature-importance.html





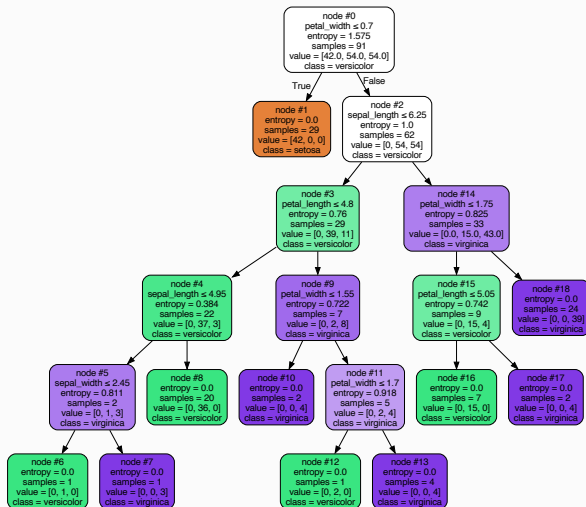
# Decision Tree # 7

**Notebook:** ensemble-feature-importance.html



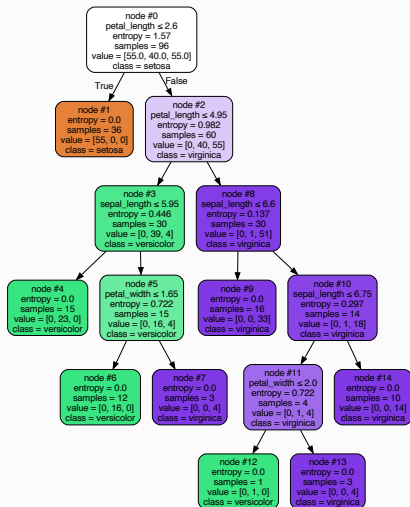
# Decision Tree # 8

## Notebook: ensemble-feature-importance.html

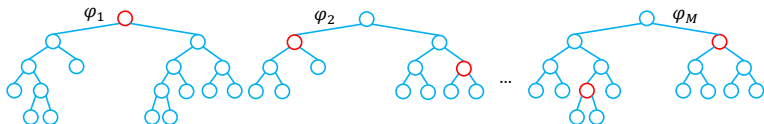


# Decision Tree # 9

## Notebook: ensemble-feature-importance.html



# Feature Importance<sup>1</sup>



Importance of variable  $X_j$  for an ensemble of  $M$  trees  $\varphi_m$  is:

$$\text{Imp}(X_j) = \frac{1}{M} \sum_{m=1}^M \sum_{t \in \varphi_m} 1(j_t = j) \left[ p(t) \Delta i(t) \right],$$

where  $j_t$  denotes the variable used at node  $t$ ,  $p(t) = N_t/N$  and  $\Delta i(t)$  is the impurity reduction at node  $t$ :

$$\Delta i(t) = i(t) - \frac{N_{t_L}}{N_t} i(t_L) - \frac{N_{t_R}}{N_t} i(t_R)$$

---

<sup>1</sup>Slide Courtesy Gilles Louppe

# Computed Feature Importance

**Notebook:** ensemble-feature-importance.html

